FPGA

(CAMPO DE MATRIZ DE PUERTAS PROGRAMABLES).

1 FIELD PROGRAMMABLE GATE ARRAY

Un FPGA (Field Programmable Gate Array) permite implementar cualquier circuito digital de aplicación específica. Las aplicaciones donde más comúnmente se utilizan los FPGA incluyen procesamiento digital de señales, sistemas aeroespaciales y de defensa, sistemas de imágenes para medicina, sistemas de visión para computadoras, reconocimiento de voz, bioinformática, emulación de hardware de computadora, prototipos de ASICs entre otras. Cabe notar que su uso en otras áreas es cada vez mayor, sobre todo en aquellas aplicaciones que requieren un alto grado de paralelismo. Se estima que un FPGA supera 500 veces o más el rendimiento de un DSP, por lo tanto son usados en sistemas en tiempo real o sistemas que requieren una altísima velocidad de procesamiento.

1.1 Arquitectura de un FPGA

Un FPGA está formado por una matriz de bloques lógicos configurables (*CLB*), a su vez cada *CLB* esta formado por *Slices* y cada *Slice* esta formado por Celdas Lógicas (*Logic Cells*).

1.1.1 Celdas Lógicas/Elementos lógicos

La unidad mas pequeña de un FPGA es la celda lógica (xilinx) o elemento lógico (Altera). Una celda lógica contiene principalmente una LUT de 4 entradas (la cual se puede usar como una RAM de 16x1, o un registro de corrimiento de 16 bits), un multiplexor y un registro como se muestra en la Figura 1-1.

El registro se puede configurar como flip-flop (activo por flanco) o como latch (activo por nivel). Se pueden configurar las polaridades del reloj (Clock), habilitación (Clock enable) y señales set/reset.

1.1.2 Slices

Un Slice está formado por 2 o más celdas lógicas individuales que comparten la misma señal de reloj. Vease la Figura 1-2.

1.1.3 CLBs y LABs

Un paso arriba en la jerarquía tenemos lo que Xilinx llama Bloque lógico configurable (CLB), y Altera llama Bloque de arreglo lógico (LAB).

Algunos FPGAs de Xilinx tienen 2 slices en cada CLB y algunos tienen 4, como se muestra en la Figura 1-3.La razón para tener este tipo de arquitectura jerárquica es que es complementada por una arquitectura equivalente en la conexión. De tal forma que se tienen conexiones muy rápidas (retardos de propagación muy cortos) entre Elementos Lógicos dentro de un Slice, luego conexiones un poco mas lentas entre Slices dentro de un CLB y conexiones mas lentas entre CLBs. La finalidad es lograr una óptima compensación entre

hacer que las conexiones sean fáciles sin incurrir mucho en retardos de interconexión.

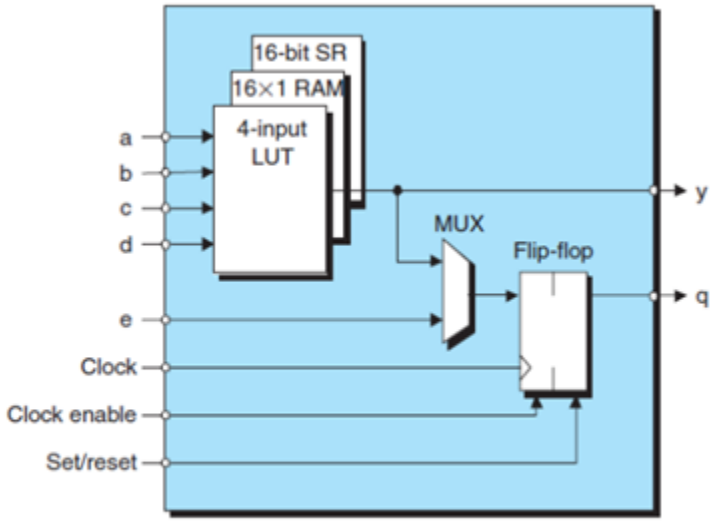

Figura 1-1. Logic Cell / Logic Element

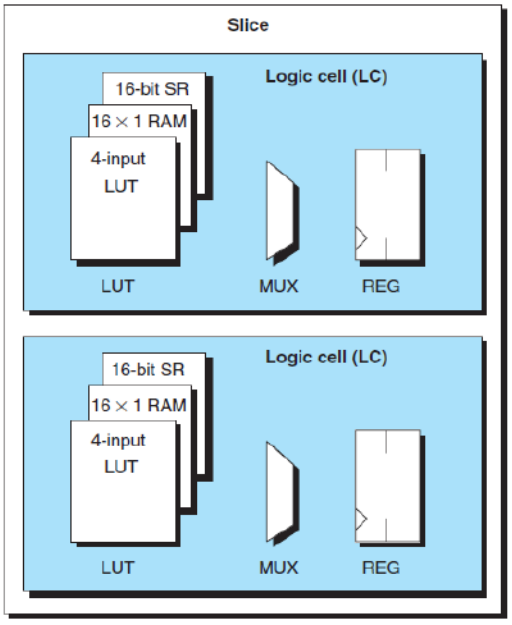

Figura 1-2. Slice

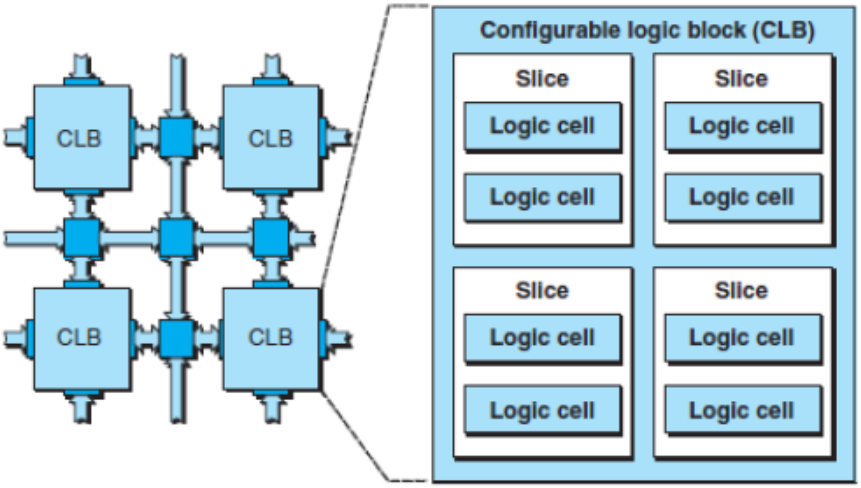

Figura 1-3. Bloques lógicos configurables (CLBs)

1.1.4 Memoria RAM distribuida y Registros de corrimiento.

Cada LUT puede ser usada como una RAM de 16 x 1, y asumiendo 4 slices por CLB como se muestra en la Figura 1-3, todas las LUTs dentro de un CLB pueden ser configuradas juntas para implementar alguna de las siguientes opciones:

- Single-port 16 x 8 bit RAM, Single-port 32 x 4 bit RAM
- Single-port 64 x 2 bit RAM, Single-port 128 x 1 bit RAM
- Dual-port 16 x 4 bit RAM, Dual-port 32 x 2 bit RAM
- Dual-port 64 x 1 bit RAM

Además cada LUT de 4 bits puede ser usada como un registro de desplazamiento de 16 bits. En este caso, existen conexiones especiales entre las celdas lógicas dentro de un slice y entre slices también para permitir que el último bit de un slice se conecte al primero de otro slice permitiendo implementar registro de corrimiento de 128 bits por CLB.

La Tabla 1-1 muestra la cantidad de CLBs, Slices, LUTs y memoria RAM distribuida de la familia Spartan 3 de Xilinx. Cada celda lógica también contiene lógica especial para manejo de acarreos, la cual permite realizar circuitos aritméticos tales como contadores. Juntando está lógica acarreos con los registros de desplazamiento y multiplicadores embebidos permite a los FPGA realizar operaciones de procesamiento digital de señales.

Tabla 1-1. Recursos de la familia Spartan-3

Device	CLB Rows	CLB Columns	CLB Total	Slices	LUTs / Flip-Flops	Equivalent Logic Cells	RAM16 / SRL16	Distributed RAM Bits
Spartan®-3A DSP FPGA CLB Resources								
XC3SD1800A	88	48	4,160	16,640	33,280	37,440	16,640	266,240
XC3SD3400A	104	58	5,968	23,872	47,744	53,712	23,872	381,952
Spartan-3A/3AN FPGA CLB Resources								
XC3S50A/AN	16	12	176	704	1,408	1,584	704	11,264
XC3S200A/AN	32	16	448	1,792	3,584	4,032	1,792	28,672
XC3S400A/AN	40	24	896	3,584	7,168	8,064	3,584	57,344
XC3S700A/AN	48	32	1,472	5,888	11,776	13,248	5,888	94,208
XC3S1400A/AN	72	40	2,816	11,264	22,528	25,344	11,264	180,224
Spartan-3E FPGA CLB Resources								
XC3S100E	22	16	240	960	1,920	2,160	960	15,360
XC3S250E	34	26	612	2,448	4,896	5,508	2,448	39,168
XC3S500E	46	34	1,164	4,656	9,312	10,476	4,656	74,496
XC3S1200E	60	46	2,168	8,672	17,344	19,512	8,672	138,752
XC3S1600E	76	58	3,688	14,752	29,504	33,192	14,752	236,032
Spartan-3 FPGA CLB Resources								
XC3S50	16	12	192	768	1,536	1,728	768	12,288
XC3S200	24	20	480	1,920	3,840	4,320	1,920	30,720
XC3S400	32	28	896	3,584	7,168	8,064	3,584	57,344
XC3S1000	48	40	1,920	7,680	15,360	17,280	7,680	122,880
XC3S1500	64	52	3,328	13,312	26,624	29,952	13,312	212,992
XC3S2000	80	64	5,120	20,480	40,960	46,080	20,480	327,680
XC3S4000	96	72	6,912	27,648	55,296	62,208	27,648	442,368
XC3S5000	104	80	8,320	33,280	66,560	74,880	33,280	532,480

1.1.5 Memoria RAM Embebida

Muchas aplicaciones requieren el uso de memoria RAM, de tal forma que los FPGA actuales poseen largos bloques de memoria RAM llamada *e-RAM* o *block RAM*.

Dependiendo de la arquitectura del FPGA estos bloques pueden estar posicionados alrededor de la matriz de CLBs u organizada en columnas como se muestra en la Figura1-4.
La cantidad de memoria RAM embebida varía entre dispositivos y entre familias, algunos pueden tener cientos de kilo bits y otros FPGAs más avanzados pueden tener cientos de Mega bits.

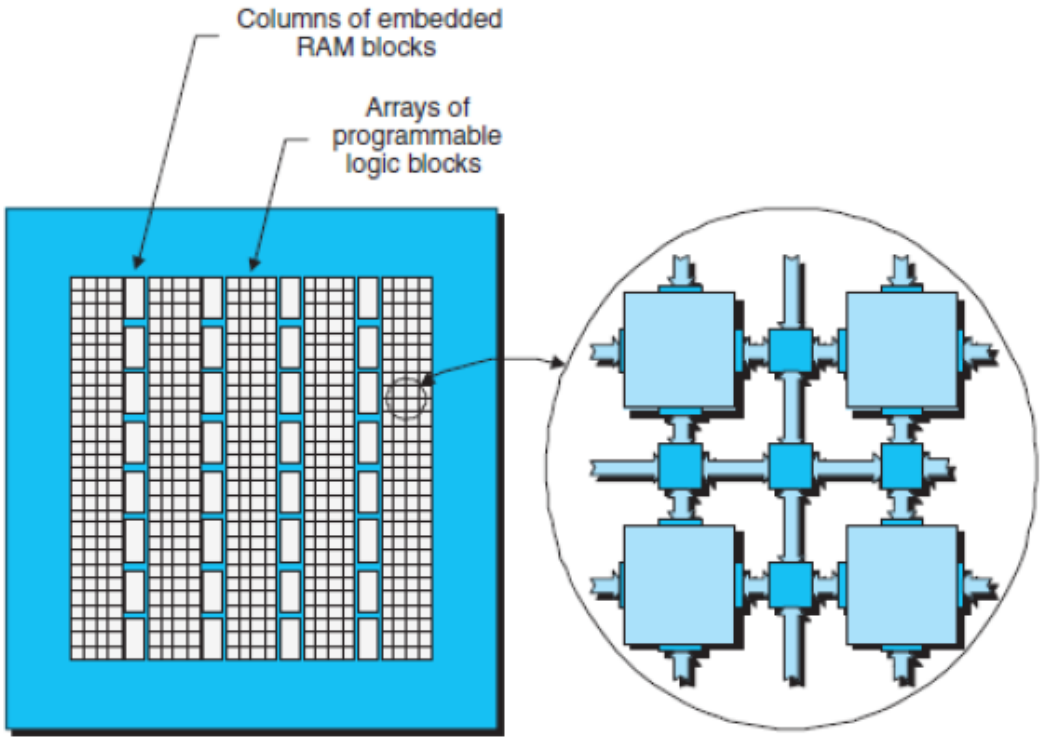

Figura 1-4. Columnas de memoria RAM embebida

La Tabla 1-2 muestra la cantidad de memoria RAM embebida de la familia Spartan-3.

Tabla 1-2. Block RAM disponible en Spartan-3

Family	Device	RAM Columns	RAM Blocks Per Column	Total RAM Blocks	Total RAM Bits	Total RAM Kbits
Extended Spartan-3A FPGAs	XC3SD1800A	4	20-22	84	1,548,288	1,512K
	XC3SD3400A	5	24-26	126	2,322,432	2,268K
	XC3S50A/AN	1	3	3	55,296	54K
	XC3S200A/AN	2	8	16	294,912	288K
	XC3S400A/AN	2	10	20	368,640	360K
	XC3S700A/AN	2	10	20	368,640	360K
	XC3S1400A/AN	2	16	32	589,824	576K
Spartan-3E FPGAs	XC3S100E	1	4	4	73,728	72K
	XC3S250E	2	6	12	221,184	216K
	XC3S500E	2	10	20	368,640	360K
	XC3S1200E	2	14	28	516,096	504K
	XC3S1600E	2	18	36	663,552	648K

1.1.6 Sumadores, Multiplicadores y MAC Embebidos.

Los circuitos sumadores y multiplicadores son lentos ya que generan largos retrasos de propagación al conectar un gran número de CLBs juntos. Debido a esto y a que son operaciones muy comunes, muchos FPGAs incorporan bloques sumadores y multiplicadores. Estos se encuentran comúnmente localizados cerca de la memoria RAM embebida debido a que comúnmente se usa en conjunto con esta.

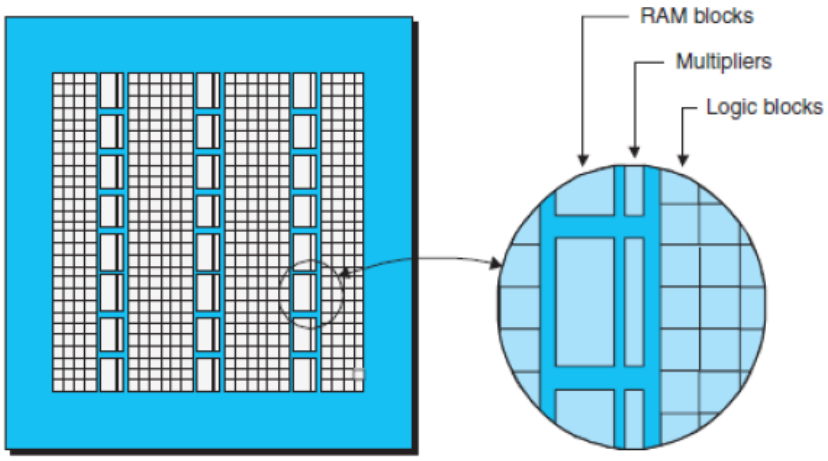

Figura 1-5. Multiplicadores embebidos

Una operación muy común en las aplicaciones de procesamiento digital de señales (DSP) es llamada *Multiplica y acumula (MAC)*. Como su nombre indica, esta función multiplica dos números y luego suma el resultado a un acumulador (Figura 1-6). Esta operación es usada para el cálculo de la convolución, FFT (transformada rápida de Fourier), FWT (transformada rápida de Wavelet), etc.

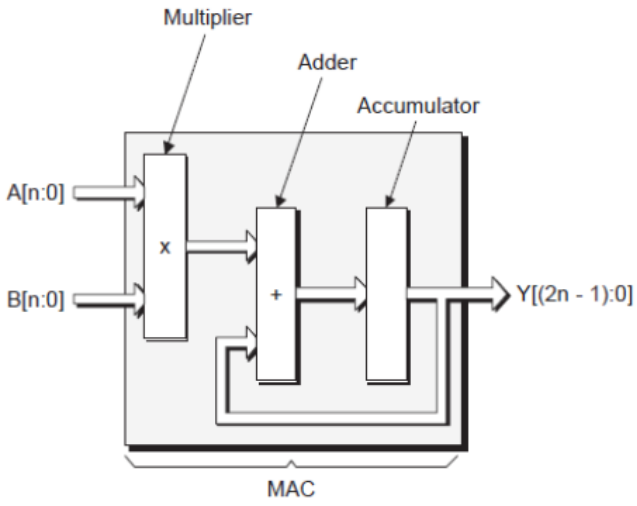

Figura 1-6. Módulo MAC

1.1.7 Procesadores Embebidos

Los FPGAs nos brindan la posibilidad de diseñar sistemas digitales de muy alta velocidad (picosegundos, nanosegundos) y un alto grado de paralelismo. Sin embargo la mayoría de las aplicaciones requieren realizar gran número de operaciones en forma de software, por lo tanto casi todos los diseños digitales hacen uso de microprocesadores o microcontroladores. Hasta hace poco, los sistemas digitales hacían uso de microprocesadores y FPGAs en la misma tablilla de circuitos de tal forma que cuando se requería acelerar un algoritmo por hardware se pasaba la tarea al FPGA y el resto de las operaciones (de menor velocidad) las realizaba el microprocesador.

Actualmente los FPGAs de la gama alta poseen uno o más microprocesadores embebidos denominados *microprocessor cores* de tal forma que un mismo FPGA puede hacer las tareas tanto por software como por hardware implementando lo que le llaman System on Chip (SoC) dando importantes ventajas como son:

• Disminución de costo (un solo chip).
• Disminución de un gran número de pistas, pines y pads en el PCB.
• Circuitos más pequeños y livianos.

De manera general existen 2 tipos de *microprocessor cores*:

• Hard Microprocessor core
• Soft Microprocessor core

Hard Microprocessor Core

Un *hard microprocessor core* viene implementado de fábrica dentro del FPGA.

Los podemos encontrar de 2 formas:

1. Localizado en una tira (llamada *"the stripe"*) a un lado de la estructura del FPGA como se muestra en la Figura 1-7. Esta implementación tiene la ventaja que la estructura del FPGA (matriz de CLBs con *e-RAM* y multiplicadores embebidos) es idéntica para dispositivos que no tienen microprocesadores embebidos, lo cual ayuda a hacer las cosas mas fáciles para las herramientas de diseño usadas por los ingenieros. La otra ventaja es que el fabricante de FPGAs puede agregar muchas funcionalidades en la tira para complementar el microprocesador, tales como memoria, periféricos, etc.

2. Lo podemos encontrar integrado dentro de la estructura del FPGA. Actualmente existen FPGAs con 1, 2 o incluso 4 microprocesadores embebidos dentro de la estructura (Figura 1-8). En este caso, las herramientas de diseño deben ser capaces de tomar en cuenta la presencia de estos bloques. Algunos prefieren esta arquitectura debido a que se pueden obtener mejorías en velocidad de transmisión debido a la cercanía con la arquitectura del FPGA.

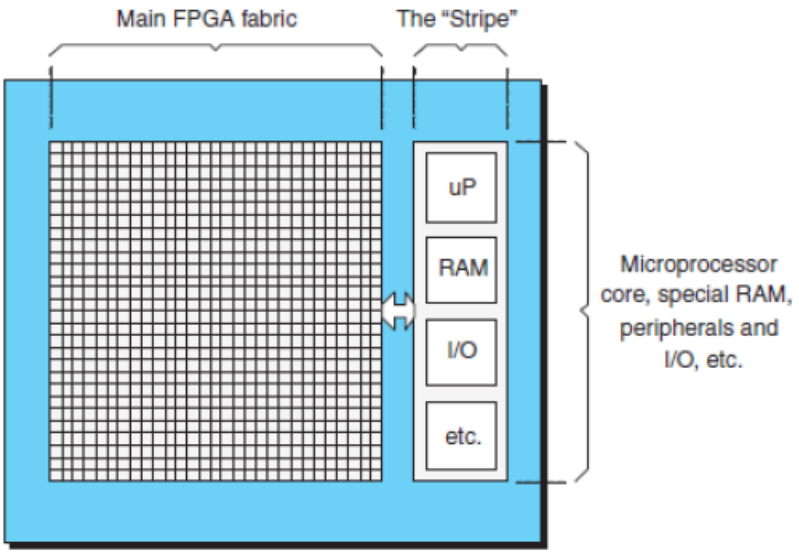

Figura 1-7. Hard microprocessor core (The stripe)

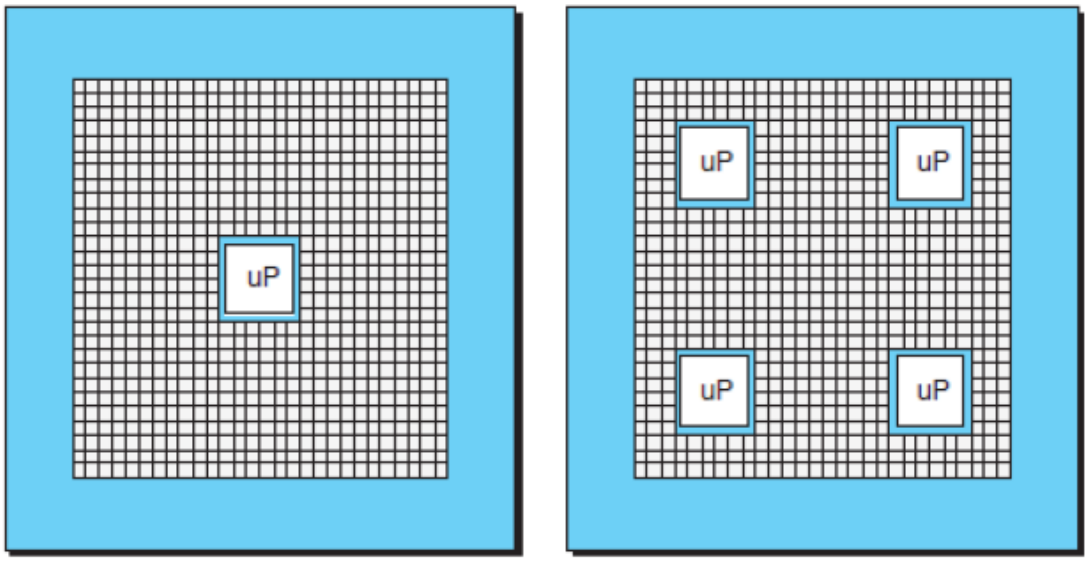

Figura 1-8. Hard microprocessor core dentro de la estructura

.

Soft Microprocessor Core

En lugar de integrar el microprocesador dentro del FPGA, es posible diseñar un microprocesador dentro del FPGA usando los CLBs. Este tipo de procesadores son llamados *soft cores*. Los *soft cores* son mas sencillos y lentos que los *hard cores* (30% a 50% la velocidad de un *hard core* [2]), sin embargo tienen la ventaja que se pueden implementar en cualquier FPGA (no tiene que ser de la gama alta) y un mismo FPGA puede tener decenas o centenas de *soft cores* funcionando en paralelo.

1.1.8 Terminales de E/S de propósito general

Los FPGAs actuales pueden tener más de mil pines acomodados como una matriz en la base del chip. La Figura 1-9 muestra el encapsulado FFG1924 de 1924 terminales.

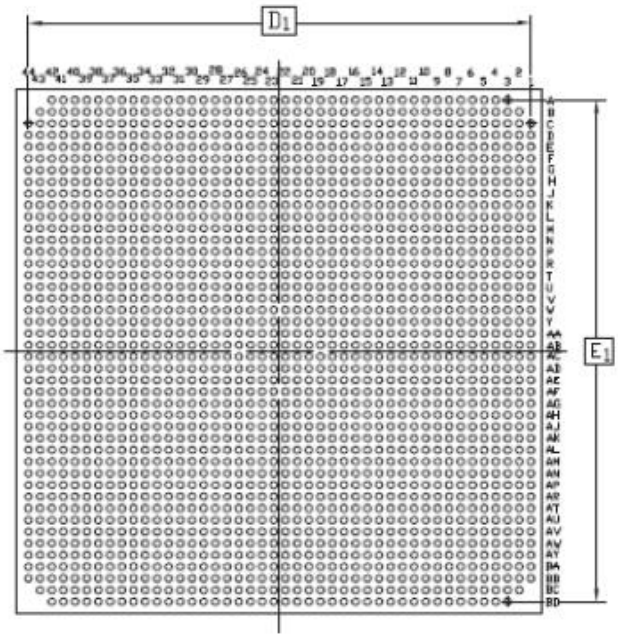

Figura 1-9. FFG1924 package

Las terminales de los FPGA se pueden configurar para aceptar y generar señales de una gran variedad de estándares (llámese por estándar los aspectos eléctricos de las señales, por ejemplo los niveles de voltaje del '0' y '1' lógico).

Estas señales de propósito general se dividen en bancos, por ejemplo en la Figura 1-10 se muestran un FPGA con 8 bancos y cada banco se puede configurar para soportar un estándar diferente.

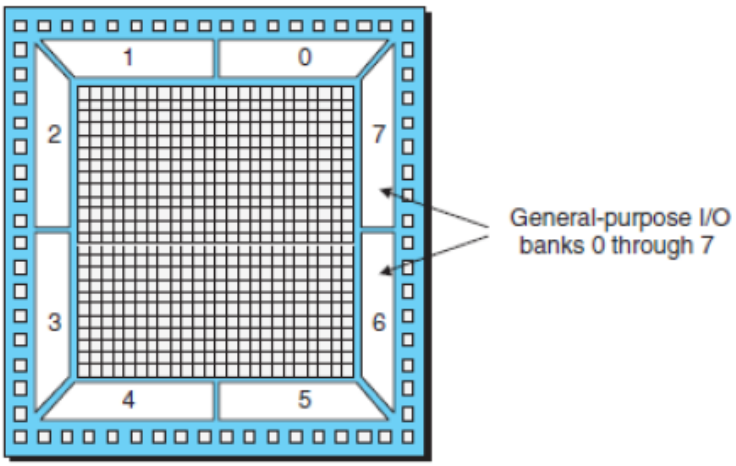

Figura 1-10. Bancos de Terminales de E/S

La Tabla 1-3 muestra los estándares Single-Ended (1 solo pin) soportados por la familia Spartan-3 de Xilinx.

Standard	V_{CCO}	Class	Spartan-3 FPGAs	Spartan-3E FPGAs	Extended Spartan-3A FPGAs
LVCMOS	1.2V	-	up to 6 mA	2 mA	up to 6 mA
	1.5V	-	up to 12 mA	up to 6 mA	up to 12 mA
	1.8V	-	up to 16 mA	up to 8 mA	up to 16 mA
	2.5V	-	up to 24 mA	up to 12 mA	up to 24 mA
	3.3V	-	up to 24 mA	up to 16 mA	up to 24 mA
LVTTL	3.3V	-	up to 24 mA	up to 16 mA	up to 24 mA
PCI33	3.0V	-	√	√	√
	3.3V	-	√	√	√
PCI66	3.0V	-		√	√
	3.3V	-		√	√
SSTL	1.8V	I	√	√	√
		II	√		√
	2.5V	I	√	√	√
		II	√		√
	3.3V	I			√
		II			√
HSTL	1.5V	I	√		√
		III	√		√
	1.8V	I	√	√	√
		II	√		√
		III	√	√	√
GTL	-	-	√		
	-	Plus	√		
DCI option	-	-	√		

Tabla 1-4. Differential I/O Standards

Standard	V_{CCO}	Spartan-3 FPGAs	Spartan-3E FPGAs	Extended Spartan-3A family FPGAs
LVDS	2.5V	√	√	√
	3.3V			√
BLVDS	2.5V	√	√	√
MINI_LVDS	2.5V		√	√
	3.3V			√
LVPECL	2.5V	√	√	√
	3.3V			√
RSDS	2.5V	√	√	√
	3.3V			√
TMDS	2.5V			
	3.3V			√
PPDS	2.5V			√
	3.3V			√
LDT	2.5V	√		
LVDSEXT	2.5V	√		
DIFF_SSTL	-	√	√	√
DIFF_HSTL	-	√	√	√
DIFF_TERM	-		√	√

Tabla 1-4. Differential I/O Standards

1.1.9 Transceptores de alta velocidad

Los FPGAs actuales de la gama alta incluyen bloques de transceptores de alta velocidad. Estos transceptores pueden operar a velocidades de Gigabits/s (hasta 6.6
Gp/s la familia Virtex-6). Algunos de los estándares de comunicación digital que soportan los FPGA son:

- Fibre Channel
- InfiniBand
- PCI Express
- RapidIO TM
- SkyRail TM (from MindSpeed Technologies)
- 10-gigabit Ethernet

1.2 Propiedad Intelectual (IP)

Los diseños actuales con FPGA son tan grandes y complejos que sería impráctico crear cada porción del diseño desde cero. Una solución a esto es reutilizar bloques funcionales existentes y dedicar la mayor parte del tiempo en crear las nuevas porciones del diseño que lo hacen diferente de otros.

A los bloques funcionales existentes se les llama Propiedad Intelectual. Existen tres fuentes principales de IPs:

1. Bloques creados previamente, reutilizados de diseños previos.
2. Fabricantes de FPGA.
3. Provedores de IPs externos.

Cada fabricante de FPGA tiene su propia colección de hard, firm y soft IP. Los Hard IPs vienen en forma de bloques preimplementados, tales como cores de microprocesador, interfaces gigabit, multiplicadores, sumadores, funciones MAC, etc.

Estos bloques están diseñados para ser lo más eficiente posible en términos de consumo de potencia, espacio y rendimiento.

Los Soft IPs vienen en forma de librerías en lenguaje de alto nivel que pueden ser incluidas en los diseños.

Los firm IPs se encuentran en un punto medio, los cuales también vienen en forma de funciones en librerías, pero a diferencia de los soft IP, estas funciones ya están óptimamente mapeadas, colocadas y ruteadas en un grupo de bloques lógicos programables.

1.3 Lenguajes de Programación para FPGAs

Existen varios lenguajes de programación para diseñar circuitos digitales para FPGAs. A continuación se listan algunos de estos.

- VHDL
- Verilog
- System Verilog
- SystemC
- Handel-C
- Pure C/C++
- Simulink
- LabVIEW

VHDL y Verilog son de los primeros lenguajes y los más utilizados para la síntesis y la simulación de sistemas digitales para FPGA.

Conforme los FPGAs fueron creciendo en potencia y velocidad, surgieron versiones de lenguajes de alto nivel conocidos como C y C++ para FPGA. Otras opciones de más alto nivel de abstracción incluyen Simulink y LabVIEW.

Aquí en este curso se trabajará con lenguaje VHDL ya que es el más utilizado y difundido.

1.3.1 VHDL

VHDL es el acrónimo que representa la combinación de **VHSIC** (Very High speed Integrated Circuit) y **HDL** (Hardware description language).

Originalmente VHDL fue patrocinado por el Departamento de Defensa de E.U. y posteriormente transferido al IEEE (Institute of Electrical and Electronics Engineers). El IEEE lo ratificó como el estándar 1076 en 1987, al cual se le llama VHDL-87. En 1993 el IEEE hizo una nueva revisión del estándar (VHDL-93) y en el 2001 (VHDL-2001).

El VHDL originalmente surgió como un lenguaje para simulación de circuitos digitales y para el diseño se usaban otras herramientas como esquemáticos y Netlist.

Conforme los circuitos digitales fueron haciéndose más complejos surgió la necesidad de poder describir los circuitos con un alto grado de abstracción, no desde el punto de vista estructural, sino desde el punto de vista funcional.[8] Este nivel de abstracción ya se había alcanzado con las herramientas de simulación (como VHDL), ya que para poder simular partes de un circuito era necesario disponer de un modelo que describiera el comportamiento del circuito o sus componentes. Fue entonces cuando se empezó a usar el VHDL para el diseño de circuitos digitales, ya que surgieron herramientas que realizan la síntesis a partir de la descripción en HDL.

El lenguaje VHDL es un lenguaje muy extenso y complejo, sin embargo para la síntesis solo se una pequeña porción del lenguaje. Es esta pequeña porción del lenguaje la que usaremos a lo largo de este curso.

1.3.2 Tarjetas de Desarrollo

Existen tarjetas de desarrollo de diversos fabricantes que contienen un FPGA y un conjunto de componentes externos como pueden ser:

botones, interruptores, leds, displays de 7 segmentos, display LCD, memoria RAM, conectores para audio, video VGA, USB, etc.

Estas tarjetas de desarrollo nos permiten realizar pruebas de manera rápida sin tener que diseñar, armar y soldar un circuito con FPGA para poder usarlo.

En este curso usaremos la tarjeta **Basys2** de **Digilent** mostrada en la Figura 1-11 para probar los diseños que se hagan durante el curso.

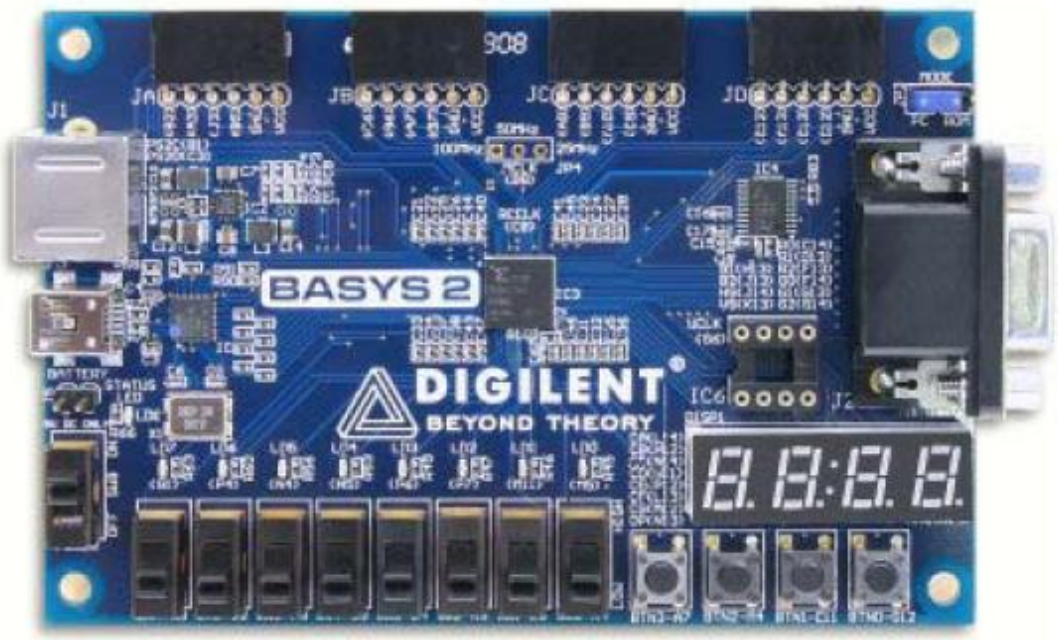

Figura 1-11. Basys2

La **Basys2** contiene lo siguiente:

• Un FPGA Spartan3E modelo XC3S100E con 240 CLBs equivalente a 100,000 compuertas lógicas.

• Un puerto USB Atmel AT90USB2 por el cual se graba y se proporciona alimentación a la tarjeta.

• Una ROM Flash para almacenar configuraciones del FPGA.

• 8 LEDs, 4 displays de 7 segmentos, 4 botones y 8 interruptores.

• 1 puerto PS/2

• 1 puerto VGA de 8 bits.

• Reloj de 25/50/100 MHz, además de un socket para otro reloj.

• Circuito de protección de corto circuito y ESD en todas las señales de E/S.